虚拟化与数字仿真融合的网络仿真任务划分 *

吴文燕 [1a]，姜　鑫 [2]，王晓锋 [1a]，刘　渊 [1b]

(1. 江南大学 a. 物联网工程学院; b. 数字媒体学院，江苏 无锡 214122; 2. 江南计算技术研究所，江苏 无锡 214083)

摘　要：为提升网络仿真性能，面向虚拟化与数字仿真融合的网络仿真体系架构，研究相应的网络仿真任务划分方法。综合考虑虚拟化与数字仿真各自优势，将网络拓扑分为虚拟化拓扑区域与数字仿真拓扑区域，结合给定物理资源，以负载均衡与远程通信量最小化为目标，对两种区域进行融合划分。实验表明，通过该方法进行网络仿真任务划分相对于随机算法与均衡负载平衡算法，远程通信量分别平均降低 33.7%，25.1%，负载均衡度分别平均提升 56.3%，38.0%。该方法可有效降低远程通信量与提升负载均衡度。

关键词：虚拟化；数字仿真；融合仿真；任务划分；远程通信量；负载均衡度

中图分类号：TP393　　doi: 10.3969/j.issn.1001-3695.2017.09.0943

Task dividing of network emulation for fusion of virtualization and digital simulation

Wu Wenyan[1a], Jiang Xin[2], Wang Xiaofeng[1a], Liu Yuan[1b]

(1. *a. School of Internet of Things Engineering*, *b. School of Digital Media*, *Jiangnan University*, *Wuxi Jiangsu* 214122, *China*; 2. *Jiangnan Institute of Computing Technique*, *Wuxi Jiangsu* 214083, *China*)

Abstract: Researched on the task dividing method based on the architecture of the network emulation for the fusion of virtualization and digital simulation to improve the performance. This method took into account the advantages of virtualization and digital simulation, and the emulation network topology was divided into virtualization topology area and digital simulation topology area, and then divided the two topology area combined with given physical resources aiming at load balancing and remote traffic minimizing. Extensive experiments showed that using the method to divide the network emulation task, the remote traffic was reduced by 33.7%, 25.1% averagely, and the degree of load balancing was improved by 56.3%, 38.0% averagely, compared with the random algorithm and the uniform load balancing algorithm. The task dividing method can effectively reduce the remote traffic and improve the degree of load balancing.

Key Words: virtualization; digital simulation; fusion emulation; task dividing; remote traffic; degree of load balancing

0　引言

网络技术研究与信息系统安全评估平台为计算机技术与网络安全评估提供了有力支撑，而网络仿真技术是整个平台的基石。但是，如何进一步提高网络研究实验中的大规模性与高逼真性仍是目前亟需解决的问题，针对此问题，虚拟化与数字仿真融合的网络仿真技术成为网络研究新的趋势。例如，文献[1]综合分布式网络模拟技术以及基于虚拟化的网络仿真技术，实现了模拟、仿真相融合的实验平台，重点研究了网络模拟与网络仿真间的同步问题；文献[2]针对网络实验中路由器仿真的规模性、扩展性以及逼真度的问题，结合虚拟化技术和数字仿真技术，提出了一种虚拟化的路由仿真平台设计方法；文献[3,4]结合虚拟化平台技术，通过将物理资源虚拟化，并将多个全功能网络节点和模拟通信链路实体部署在虚拟机层面，设计了一种网络仿真准实验测试床；文献[5]提出了一个集成了虚拟化仿真系统和并行离散事件网络模拟器的网络测试平台，设计了一个全球同步的算法来管理虚拟时间与仿真时间的转换。

面向大规模网络仿真的性能需求，如何实现合理、有效的网络仿真任务划分方法，保证待划分拓扑与计算集群间的有效映射以及在合理利用物理资源的基础上降低计算集群间的通信开销、满足负载平衡是关键。文献[6]综合考虑了具体的同构计算环境和所要模拟的网络模型，将负载平衡、通信开销与同步开销最小化作为划分目标；文献[7]重点分析了如何有效评估所要仿真的网络拓扑中各网络节点的数据包转发量，并以此作为基础，研究了分布式网络仿真平台的数据包转发模拟任务的负载平衡方法；文献[8]以流量估计为基础，研究了基于抽象消减

基金项目：国家自然科学基金资助项目（61672264）；国家重点研发计划资助项目（2016YFB0800801）

作者简介：吴文燕（1994-），女，安徽铜陵人，硕士研究生，主要研究方向为网络仿真（1837512594@qq.com）；姜鑫（1982-），男，吉林白城人，工程师，博士，主要研究方向为网络安全；王晓锋（1978-），男，江苏无锡人，副教授，博士，主要研究方向为网络仿真、网络安全；刘渊（1967-），男，江苏无锡人，教授，硕士，主要研究方向为网络安全.

的网络模拟拓扑划分方法，以此来均衡数据包转发模拟任务，减少远程链路数，并提高网络拓扑划分效率；文献[9]提出了一种基于多项式时间复杂度的负载均衡划分算法，该方法能在均衡负载的同时尽可能的减少通信消耗；文献[10]提出了一种基于虚拟拓扑预配置及可重用技术的虚拟网络映射算法以提高映射公平性，将虚拟网路映射过程分为两个步骤：拓扑预配置过程和映射过程，该算法在物理网络资源利用率、收益/成本比及虚拟网络接受公平性等方面优于部分同类算法；文献[11]为加强测试资源集中管理，结合实际中 ATF(automation test framework，自动化测试框架)拓扑映射，提出了智能云测试平台自动拓扑映射实现的方法，该拓扑映射算法扩展性及保密性较好，在发现效率、准确性、有效性上有了很大提高；文献[12]针对网络虚拟化环境中的能耗问题，根据网络拓扑属性以及底层物理网络中物理节点和网络设备的能耗特性，建立网络拓扑一致、高效节能的虚拟网络映射模型，将相邻的虚拟节点映射到能耗增幅较小的邻接物理节点上，同时协调使用最小能耗路由算法，将虚拟链路映射到能耗最小的物理路径上，该算法有效地提高了休眠物理节点和休眠物理链路数量,显著地降低了虚拟网络映射的资源代价和系统能耗。

以上网络仿真任务划分方法兼顾了物理资源利用率与仿真性能，但都是基于单一的虚拟化或数字仿真技术，并未考虑到虚拟化与数字仿真融合的网络拓扑划分情况。但是，在实际网络环境下，一个网络拓扑中往往有不止一种网络节点，进行网络仿真实验时，不同的网络节点使用不同的网络映射方式，因此融合网络拓扑划分方法就显得尤为重要。

鉴于虚拟化与数字仿真融合的网络仿真是一种趋势，本文面向基于云计算平台构建的虚拟化与数字仿真融合的网络仿真体系架构，针对性的研究了虚拟化与数字仿真融合的网络仿真任务划分方法。该方法兼顾了网络拓扑中的虚拟化与数字仿真节点，并在尽可能确保计算集群间负载平衡的基础上大大降低计算集群间的远程通信开销，提升了融合网络仿真的性能。

1 虚拟化与数字仿真融合的网络仿真分析

1.1 虚拟化与数字仿真融合的网络仿真体系架构

如图 1 所示，是基于 OpenStack 云平台的虚拟化与数字仿真融合的网络仿真体系架构。OpenStack 的基本架构包括一个控制节点，一个网络节点和若干个计算节点，控制节点负责系统的管控功能，网络节点负责提供诸如 DHCP (dynamic host configuration protocol，动态主机配置协议)服务和路由服务等网络服务，计算节点负责为虚拟主机的建立提供资源。在 OpenStack 云平台上进行面向虚拟化与数字仿真两个尺度融合的网络仿真，而实现融合网络仿真的关键在于解决虚拟化、数字仿真和 OpenStack 平台的虚实互联、路由配置与运行协议等问题。虚实互联技术可以实现虚拟化、数字仿真与 OpenStack 平台的无缝连通，路由配置与 TCP/IP 协议族则是融合仿真时数据包正常存储转发的基础。

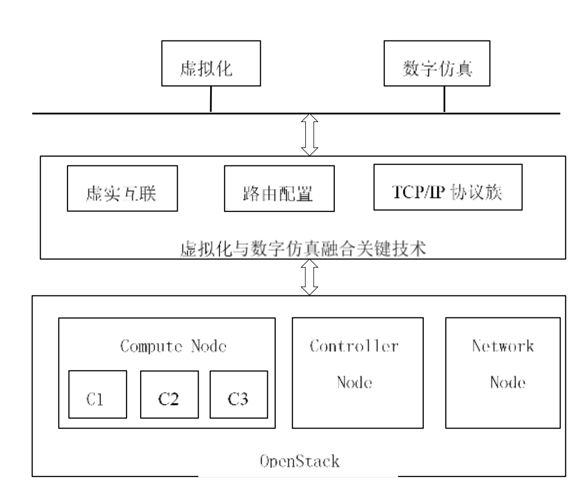


图 1 虚拟化与数字仿真融合的体系架构

1.2 虚拟化与数字仿真融合的网络仿真性能分析

虚拟化与数字仿真融合的网络仿真技术是基于 OpenStack 云平台来部署虚拟化与数字仿真融合的网络拓扑。本文中使用 KVM[13] (kernel-based virtual machine)虚拟化技术来部署虚拟化节点，KVM 在 Linux 内核部署，任何场景下都可以直接和硬件进行交互，而不需要修改虚拟化的操作系统，可以方便控制虚拟化进程；使用 NS3[14-16] 仿真技术来部署数字仿真节点的，NS3 是一个极具特色的新型网络模拟器，具有完备性、开源性、易用性和可扩展性等方面的特色，优于现有的大多数主流网络模拟器。

图 2 为在 OpenStack 云平台中实验测试 KVM 节点和 NS3 节点的吞吐量对比图，从实验结果可得出 KVM 节点的吞吐量要远远优于 NS3 节点；图 3 为一个 16G 内存，24 逻辑 CPU 配置的计算节点可以启动的 KVM 节点（4G 内存，2 逻辑 CPU）与 NS3 节点数量对比图，从实验结果可以得出 NS3 节点的规模性远远优于 KVM 节点。

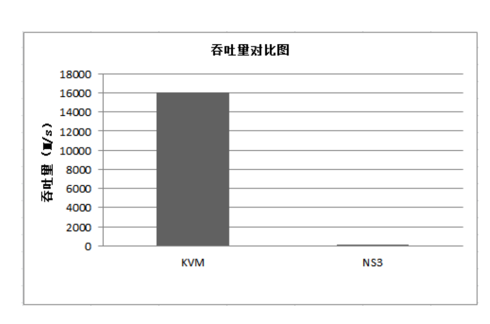


图 2 吞吐量对比图

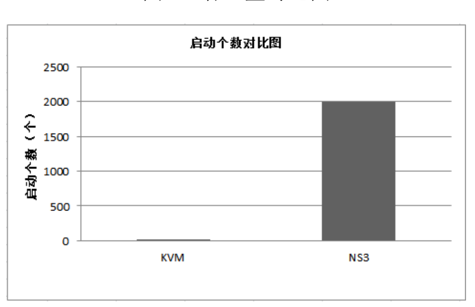


图 3 启动个数对比图

总的来说，KVM 吞吐量高，但由于 KVM 节点在保证正常工作情况下的启动个数受计算节点内存大小和逻辑 CPU 个数限制，所以一个计算节点能够启动的 KVM 节点数比较少，成个数级；而 NS3 吞吐量低，在保证正常工作情况下的启动个数虽然也受计算节点逻辑 CPU 个数的限制，但是计算节点的内存大小对它的启动个数几乎没有影响，一个高性能计算节点上能同时运行数千个 NS3 仿真节点。所以，KVM 逼真度优于 NS3，但是规模性是 NS3 优于 KVM，因此面向虚拟化与数字仿真融合的网络仿真技术，可有效解决仿真网络的逼真性和可伸缩性需求。

1.3　融合的网络仿真任务划分问题描述

虚拟化与数字仿真融合的网络仿真任务划分方法充分考虑了不同节点间的差异性与不同节点组成的链路间通信量的需求，在尽可能节约物理资源的基础上，从计算环境整体角度，综合考虑计算节点的负载均衡与计算集群间的远程链路通信量，以此为依据进行网络仿真拓扑的快速准确划分。

一个网络拓扑中会有不同的节点类型，总体可分为路由节点和主机节点，而路由节点又可分为终端路由节点和非终端路由节点，网络拓扑中的终端路由节点在去掉与主机节点相连的链路以后，只与拓扑中另外一个节点相连，即度为 1。一个网络拓扑中，不同类型的节点可以使用不同的映射方式，本文算法主要从吞吐量与规模性两方面来考虑不同类型的节点适用的映射方式。

基于 KVM 节点和 NS3 节点的优缺点，本文提出了虚拟化与数字仿真融合的网络仿真任务划分方法。根据实际网络情况可知，网络拓扑中的终端路由节点和主机节点的通信量较小，负载也较小，而非终端路由节点的通信量较大，负载也较大，且主机节点数目远远超过路由节点数目；基于上述特点，将所要划分的网络拓扑中的终端路由节点与主机节点采用 NS3 进行映射，将非终端路由节点采用 KVM 进行映射。

2　虚拟化与数字仿真融合网络仿真任务划分方法

2.1　分布式网络仿真任务划分

网络仿真任务划分的原理是将网络仿真拓扑分成多个子网，每个子网的仿真任务分别交给不同的计算节点来执行，而各子网之间的通信则通过计算节点之间的真实链路实现。如图 4 所示，是一个简单的分布式网络仿真任务划分模型，描述了同构计算环境下的网络仿真任务划分问题。图 4 中 N 为所要仿真的网络拓扑中的拓扑节点总数，W 为所要仿真的网络拓扑中的各拓扑节点负载之和(一个拓扑节点的负载为该拓扑节点的权值)。设同构计算环境中共有 M 个计算节点，分别用计算节点 1~M 表示。在总网络拓扑划分后，计算节点 1~M 所分配到的拓扑节点个数分别为 n1~nM，所分配到的拓扑节点的负载之和分别为 w1~wM。其中每个计算节点所分配到的拓扑节点个数反映计算节点被分配的路由仿真任务量，每个计算节点所分配到的拓扑节点负载之和反映计算节点被分配到的数据包转发仿真任务量。

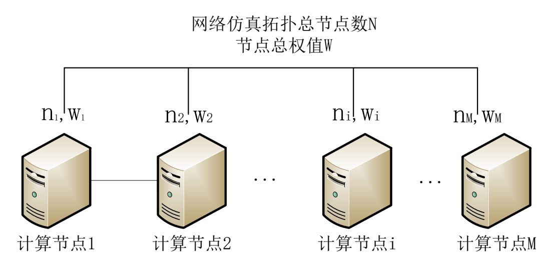


图 4　分布式网络仿真任务划分模型

本文中待划分的网络仿真拓扑，非终端路由节点由虚拟化节点映射，终端路由节点和主机节点由数字仿真节点映射。并且虚拟化与数字仿真融合的网络仿真任务划分的目的是使整个计算环境中负载平衡与通信开销最小，这里的通信开销主要是指远程链路之间的通信开销，因为在分布式网络仿真中，网络拓扑被划分为一些小的拓扑，原来的网络拓扑之间的链路被划分开并作为远程链路分派到不同的计算节点上，这样数据包在转发时的开销就会显著增加。

2.2　虚拟化与数字仿真融合的网络仿真任务划分算法

虚拟化与数字仿真融合的网络仿真任务划分算法步骤如下，划分时只划分终端路由节点和非终端路由节点，主机节点划分在与之相连的终端路由节点的划分结果中。

假设有一网络拓扑，其中共有 N 个路由节点，则分别编号为 1~N，创建矩阵 Topo[N][N]用来存储网络拓扑相对负载值信息。例如，编号为 i 与编号为 j 的路由节点之间若有链路，则将此链路相对负载值存储在 Topo[i-1][j-1]中；若无链路，则 Topo[i-1][j-1]的值为 0；Topo[i-1][i-1]存储的则是编号为 i 的路由节点的相对负载值，网络拓扑中路由节点的相对负载值为与此路由节点相连的链路负载值之和。

输入：赋权无向图：G(V, E)，其中点集合：V={v1,…, vj, …, vN}，点权值：wv(vj)，边权值：we(vi, vj)；

终端路由节点个数为 Z；

非终端路由节点个数为 F；

计算节点的计算能力（数据包转发模拟能力）：K；

计算节点的虚拟承受能力（能够启动的虚拟机数）：Q；

计算节点个数（子网数）：M；

输出：划分结果{G1,…, Gi,…, GM}。

算法步骤：

初始化矩阵 Topo[N][N]，Topo[i-1][j-1]存储编号为 i 和编号为 j 的路由节点之间链路的边权值，Topo[i-1][i-1]存储编号为 i 的路由节点的点权值；

遍历整个网络拓扑，找到拓扑中的终端路由节点，记录 Z 值，并将终端路由节点的点权值设为 1，再将连接终端路由节点的边的边权值设为 1，最后记录到矩阵 Topo[N][N]中，确保 METIS 初始划分时，终端路由节点的相对负载值不影响非终端路由节点的划分结果；

遍历整个网络拓扑，找到拓扑中的非终端路由节点，记录 F 值，对于每个非终端路由节点，将与它相连接的每条边的边

权值记录到矩阵 Topo[N][N]中，边权值之和为该路由节点的点权值，也记录到矩阵 Topo[N][N]中；

得出拓扑图划分个数即计算节点个数；

编号为 i 的路由节点的点权值为 Topo[i-1][i-1]= $\sum_{j=0}^{N-1} Topo[i-1][j]$；

A= $\sum_{i=0}^{N-1} Topo[i][i]$ /K，A 取整数，若有小数，则去小数再加 1；

B=F/ Q，B 取整数，若有小数，则去小数加 1；

拓扑图的划分个数为 M=max{A,B}；

初始化矩阵 R[M][N]，R[M][N]记录划分结果；

基于 METIS 获得初始划分，METIS 能基于矩阵 Topo[N][N]中的数据，并根据计算节点的数据包转发模拟能力 K，计算节点个数 M，获得数据包转发模拟任务负载较均衡以及远程通信开销较小的初始划分，设置 loop=0；

进行检验，如果每个计算节点上的路由节点权值之和不超过计算节点的计算能力 K（阈值 1）且每个计算节点上的路由节点数量小于等于一个计算节点上最多可以启动的虚拟化节点数量 Q（阈值 2），将划分结果输入到 R[M][N]中，然后进行步骤 8；如果以上两个条件有任何一个条件没有满足，则进行步骤 9；

遍历拓扑，将终端路由节点划分到与该终端路由节点相连的非终端路由节点所在的划分结果中，进行步骤 16；之所以这样做是因为终端路由节点的负载较小，不会对计算节点的负载之和造成很大影响导致负载不均衡或超过 K 值（即阈值 1），且终端路由节点为数字仿真节点，不受 Q 值（即阈值 2）的限制，还可以降低计算集群间的远程通信开销；

将路由节点权值之和不超过阈值 1、路由节点数量等于阈值 2 的计算节点的划分结果输入到 R[M][N]中，符合条件的划分结果中的路由节点的点权值和与之相连的边的边权值都设为 1，记录到 Topo[N][N]。如果 Topo[N][N]中有路由节点与 R[M][N]中的路由节点有相连边，则相连边的边权值随之变成 1，并重新计算该路由节点的点权值；如果 Topo[N][N]的值有新的变化，说明有新的符合条件的划分结果，返回步骤 6 再次划分，否则进行步骤 10；

loop=loop+1，返回步骤 6，直到 loop=5，即进行了 5 次划分结果无变化的划分且还有不符合条件的划分结果，则转入步骤 11，进行微调；

选取超过阈值（任意一个阈值）的划分子网上的拓扑节点集 Vc，该集合中的节点必须与待接收（低于阈值（所有阈值））计算节点中的某个节点之间存在链路，如果没有链路存在，则 Vc 为超过阈值的划分子网上的所有拓扑节点组成的集合；

针对 Vc，选取其中点权值最小的拓扑节点，并构成集合 V'c（点权值最小的拓扑节点可能有多个），之所以选取点权值最小的拓扑节点，是因为要使得拓扑节点迁移对数据包转发模拟任务负载平衡的影响最小；

选取 V'c 中与待迁移（超过阈值）划分子网中其他拓扑节点之间链路权值和最小的拓扑节点（如果有多个满足条件的节点则随机选取其中的一个）作为最合适移动的节点 vm，之所以这么选择是因为要使得拓扑节点迁移后新增的远程通信开销最小。

选取 vm 进行节点迁移；

返回步骤 7 进行检验；

输出划分结果{G1,…,Gi,…,GM}。

3 实验验证与分析

实验环境：一套 OpenStack 平台，包含 1 个控制节点，1 个网络节点，8 个计算节点，计算节点均为服务器 R730， CPU 类型为 2 个 Intel E5620 处理器，且均为 24 逻辑 CPU，16G 内存配置，计算节点的吞吐量均为 22000M/s；

拓扑场景：具体的，通过拓扑生成工具，生成 3 组网络拓扑。第一组网络拓扑共有 2026 个网络节点，其中 26 个为路由节点，编号为 1~26，路由节点中非终端路由节点为 22 个，终端路由节点为 4 个，并且该拓扑中每一个终端路由节点上均连接 500 个主机节点；第二组网络拓扑中共有 4027 个网络节点，其中 27 个为路由节点，编号为 1~27，路由节点中非终端路由节点为 19 个，终端路由节点为 8 个，拓扑中每一个终端路由节点上均连接 500 个主机节点；第三组网络拓扑中共有 2530 个网络节点，其中 30 个为路由节点，编号为 1~30，路由节点中非终端路由节点为 25 个，终端路由节点为 5 个，拓扑中每一个终端路由节点上均连接 500 个主机节点。基于上述实验环境，将生成的 3 组网络拓扑分别部署在此同构计算环境下。

本文算法是基于虚拟化与数字仿真融合的网络仿真任务划分算法，且由于该算法的独特性，主机节点对远程链路通信量没有影响，所以对网络拓扑的划分落实在对拓扑中路由节点的划分上。

算法 1 均衡负载平衡算法 ULB

针对给定的计算环境，METIS 划分能获得数据包转发仿真任务与路由仿真任务负载较均衡的划分结果，将面向数据包转发仿真任务与路由仿真任务负载均衡的 METIS 划分算法称为 ULB 算法；

定义 1 远程链路通信量 RLT。用于评判网络拓扑仿真运行时，各计算节点数据包转发仿真任务远程链路通信量的情况，RLT 值越小，则表明依据算法得到运行结果的计算节点的远程链路通信量越小。

定义 2 总远程链路通信量 TRLT。各计算节点远程链路通信量的累加和为计算集群的总远程链路通信量 TRLT，TRLT 值越小，则表明依据算法得到运行结果的计算集群的总远程链路通信量越小。

定义 3　算法负载均衡度 DLB。用于评判网络拓扑仿真运行时，各计算节点路由仿真任务综合负载均衡情况，该值越趋近于 0，则表明该计算环境路由仿真任务负载越均衡。用如下公式描述：

$$DLB=\sum_{i=1}^{n}\frac{1}{\text{n}}(x_i-\bar{x})^2; \quad (1)$$

因为实验拓扑是虚拟化与数字仿真融合的网络拓扑，且数字仿真节点的吞吐量远远低于虚拟化节点的吞吐量，所以本文主要考虑虚拟化路由节点对路由仿真任务负载均衡的影响，因此公式中，n 为使用到的计算集群规模，x_i 为各计算节点上所部署的虚拟化路由节点个数，$\bar{x}$ 为计算节点上所部属的虚拟化路由节点个数的期望值。由负载均衡度公式计算得出的数值越小，负载均衡度越高，反之亦然。

基于给定实验环境和网络拓扑使用虚拟化与数字仿真融合的网络仿真任务划分算法进行拓扑划分，依据算法划分结果，在 Opentack 平台上实施部署。为了验证部署在 Openstack 平台上的虚拟化与数字仿真融合的网络拓扑确实可以如真实的网络拓扑一样通信，用融合网络拓扑中的不同主机互相执行 ping 命令，结果显示融合网络拓扑中的主机可以互相 ping 通。从测试结果来看，本文提出的算法可以快速划分给定的融合网络拓扑，且根据算法得出的划分结果，可以将融合网络拓扑高逼真度地部署在 Openstack 平台上。

将融合算法的运行结果的总远程链路通信量与随机算法的划分结果以及 ULB 算法的划分结果进行比较。三组网络拓扑依据融合算法和随机算法、ULB 算法得到的划分结果的计算集群的总远程链路通信量对比如图 5 所示，其中，第一组网络拓扑通过融合算法得出的划分结果的 TRLT=16206，随机算法得出的划分结果的 TRLT=30064，ULB 算法得出的划分结果的 TRLT=23230，第二组网络拓扑通过融合算法得出的划分结果的 TRLT=12296，随机算法得出的划分结果的 TRLT=19646，ULB 算法得出的划分结果的 TRLT=17368，第三组网络拓扑通过融合算法得出的划分结果的 TRLT=27570，随机算法得出的划分结果的 TRLT=33634，ULB 算法得出的划分结果的 TRLT=33010。经计算得出融合算法相对于随机算法与 ULB 算法，划分结果的总远程链路通信量分别平均降低 33.7%，25.1%。

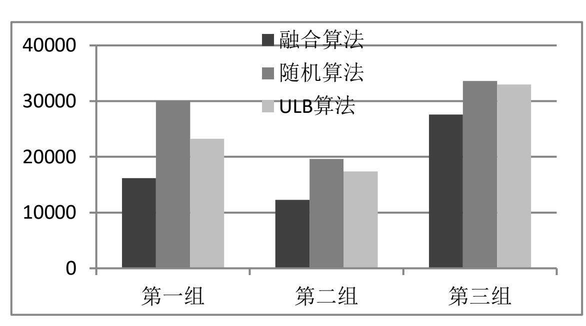


图 5　总远程链路通信量对比图

由实验结果可知，融合算法是优化的网络任务划分方法，可以减少计算集群间的总远程链路通信量。图 6~8 分别是三组网络拓扑依据不同算法得到的划分结果的各个计算节点的远程链路通信量对比图。从图 5 与图 6~8 的对比可知，通过融合算法进行划分得出的结果，虽然各个计算节点的远程链路通信量不一定小于随机算法划分结果或 ULB 算法划分结果，但是计算集群的总远程链路通信量明显小于后两者。而且，因为本文算法是虚拟化与数字仿真融合的网络仿真任务划分算法，依靠该算法划分得出的结果可以节省计算资源，如图 6 所示，随机算法划分结果以及 ULB 算法划分结果均比本文算法划分结果多出 1 个计算节点，如图 7 所示，随机算法划分结果以及 ULB 算法划分结果均比本文算法划分结果多出 2 个计算节点，如图 8 所示，随机算法划分结果以及 ULB 算法划分结果均比本文算法划分结果多出 1 个计算节点。

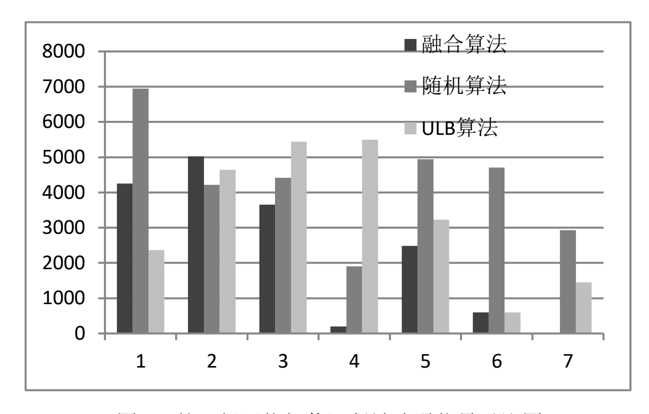


图 6　第一组网络拓扑远程链路通信量对比图

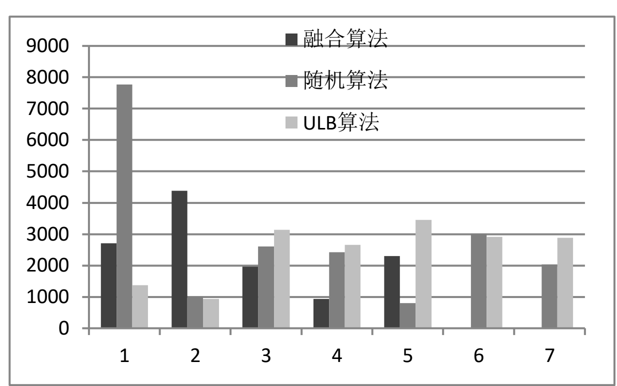


图 7　第二组网络拓扑远程链路通信量对比图

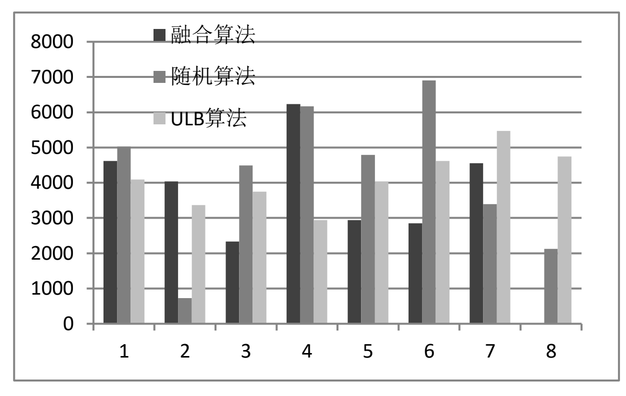


图 8　第三组网络拓扑远程链路通信量对比图

除了远程链路通信量，本文还就计算集群的负载均衡度进

行了计算。第一组网络拓扑通过融合算法得出的划分结果的DLB=0.22，随机算法得出的划分结果的 DLB=0.41，ULB 算法得出的划分结果的 DLB=0.39，第二组网络拓扑通过融合算法得出的划分结果的 DLB=0.16，随机算法得出的划分结果的DLB=0.58，ULB 算法得出的划分结果的 DLB=0.37，第三组网络拓扑通过融合算法得出的划分结果的 DLB=0.53，随机算法得出的划分结果的 DLB=1.08，ULB 算法得出的划分结果的DLB=0.61，对比如图 9 所示。经计算得出融合算法相对于随机算法与ULB算法，划分结果的负载均衡度分别平均提升56.3%，38.0%。

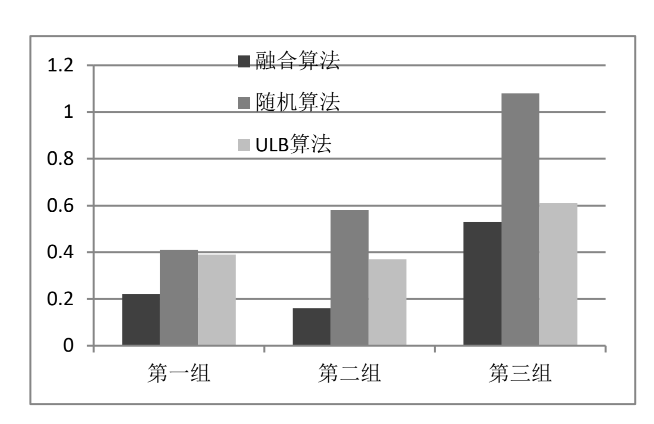


图 9　负载均衡度对比图

虚拟化与数字仿真融合的网络仿真任务划分算法的优越性在于，不但将 METIS 划分结果通过节点迁移进行优化，降低了融合算法的负载均衡度；而且由于终端路由节点划分在与终端路由节点相连接的非终端路由节点所在的划分结果中，不存在终端路由节点对远程链路通信量的影响，降低了融合算法的远程链路通信量。此外，依据虚拟化与数字仿真融合的网络仿真任务划分方法对以上 3 组网络拓扑进行划分所用的运行时间均小于 1 s。由此可见算法的时间复杂度并不高，可以适用于更大规模的网络拓扑。

4　结束语

网络仿真是当前进行网络研究的主要方法，虚拟化与数字仿真融合的网络仿真方法是网络研究新的趋势。本文重点研究了虚拟化与数字仿真融合的网络仿真任务划分算法，该算法能在节约计算资源的基础上有效降低计算集群间的远程链路通信量、提升计算集群的负载均衡度。在下一步的研究中，将重点考虑虚拟化与数字仿真融合的网络仿真任务划分方法在网络安全实验中的应用。

参考文献：

[1] Jin Du, Zheng Yi, Nicol D M. A parallel network simulation and virtual time-based network emulation testbed [J]. Journal of Simulation, 2014, 8 (3): 206-214.

[2] 黄敏桓，张尧学，许飞，等. 基于虚拟化技术的路由仿真实验平台设计 [J]. 系统仿真学报, 2014, 26 (8): 1672-1677.

[3] 黄锦松，杨艺，王文鼎. 一种基于虚拟化平台的网络仿真准实验床 [J]. 计算机技术与发展, 2015, 08 (9): 208-212.

[4] 何新华，金国柱，王琮. 仿真实验中的虚拟化技术应用 [J]. 兵器装备工程学报, 2011, 32 (8): 71-73.

[5] Gu Yun, Fujimoto R. Applying parallel and distributed simulation to remote network emulation [C]// Proc of Winter Simulation Conference. 2007: 1328-1336.

[6] Xu Ding, Ammar M. BencHMAP: Benchmark-based, hardware and model-aware partitioning for parallel and distributed network simulation [C]// Proc of International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems. Washington DC: IEEE Computer Society, 2004: 455-463.

[7] Willinger W, Taqqu M S, Sherman R, et al. Self-similarity through high-variability: statistical analysis of ethernet LAN traffic at the source level [J]. IEEE//ACM Trans on Networking, 1997, 5 (1): 71-86.

[8] Peschlow P, Honecker T, Martini P. A flexible dynamic partitioning algorithm for optimistic distributed simulation [C]// Proc of International Workshop on Principles of Advanced and Distributed Simulation. 2007: 219-228.

[9] Grande R E D, Boukerche A, Ramadan H. Distributed re-arrangement scheme for balancing computational load and minimizing communication delays in HLA-based simulations [J]. Concurrency & Computation Practice & Experience, 2013, 25 (5): 626–648.

[10] 王聪，苑迎，彭三城，等. 基于拓扑预配置的公平虚拟网络映射算法 [J]. 计算机研究与发展, 2017, 54 (1): 212-220.

[11] 王亮，韩连钢，谢锡海. 智能云测试下拓扑映射算法实现的研究 [J]. 电子技术应用, 2017, 43 (3): 116-119.

[12] 彭利民. 拓扑一致性绿色虚拟网络映射算法 [J]. 小型微型计算机系统, 2016, 37 (5): 1079-1083.

[13] Qumranet A K, Qumranet Y K, Qumranet D L, et al. kvm: the Linux virtual machine monitor [J]. Proc Linux Symposium, 2012, 10 (7): 104-112

[14] Kumar S, Paul S, Amar A K. Communication in vehicular cloud network using ns3 [J]. International Journal of Control Theory & Applications, 2017, 12 (4): 159-167.

[15] Riley G F, Henderson T R. The ns-3 network simulator [J]. Modeling & Tools for Network Simulation, 2010, 20 (5): 15-34.

[16] Font J L, Iñigo P, Domínguez M, et al. Analysis of source code metrics from ns-2 and ns-3 network simulators [J]. Simulation Modelling Practice & Theory, 2011, 19 (5): 1330-1346.